IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 9, 2019, accepted January 28, 2019, date of publication February 13, 2019, date of current version March 5, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2899101

An Experience-Based Scheme for Energy-SLA Balance in Cloud Data Centers

XIJIA ZHOU[1], KENLI LI[1], (Senior Member, IEEE), CHUBO LIU[1], AND KEQIN LI[2], (Fellow, IEEE)

[1]College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China
[2]Department of Computer Science, State University of New York at New Paltz, New Paltz, NY 12561, USA

Corresponding author: Kenli Li (lkl@hnu.edu.cn)

ABSTRACT The proliferation of cloud computing has resulted in the establishment of large-scale data centers containing thousands of computing nodes and consuming enormous amounts of electrical energy. However, the low-cost and high-efficiency slogans are getting louder and louder, and the IT industry is also striving for this pursuit. Therefore, it is vital to minimizing the energy consumption for cloud providers while ensuring the quality of service for cloud users. In this paper, we propose several heuristic strategies to optimize these two metrics based on a two-level management model under a heterogeneous cloud environment. First, to detect whether a physical node is continuously overloaded, we propose an empirical forecast algorithm, which predicts the future state of the host by statistically analyzing the historical utilization data of the host. Second, we propose a weighted priority virtual machine (VM) selection algorithm. For each VM on the overloaded host, we weight several utilization factors and calculate its migration priority. Then, we simulate the proposed approach and compare it with the existing overloaded hosts detection algorithms with different VM selection policies under different workloads.

INDEX TERMS Cloud computing, CloudSim, empirical forecast, energy consumption, quality of service, weighted priority.

I. INTRODUCTION

A. MOTIVATION

Cloud computing is a large-scale computing paradigm that provides services to customers on a ''pay-as-you-go'' basis [1]. The cloud data center provides users with various resources on demand. To support the growing demand of cloud users, cloud data centers are equipped with more and more servers to provide convenience. Since the power supply is the lifeblood of data centers, this has led to an increasing electricity consumption in data centers year by year. In addition, most of the energy is converted into heat, the demand for electrical energy for cooling in the operating environment is also increasing [2]–[4]. A survey conducted in 2016 predicts that global energy consumption will increase by nearly 50% in the next 20 years [5], and information and communication technology (ICT) accounts for a large proportion, of which data center consumes about 1.5% – 2.0% [3], [4].

The associate editor coordinating the review of this manuscript and approving it for publication was Bora Onat.

What's more, servers in data center have a narrow range of dynamic power consumption, even if completely idle servers still consume 70% of their peak power [6], [7], which causes energy waste. From a cloud service provider's perspective, it is unwise to keep a server in a low utilization state for a long time. On the one hand, a large amount of electricity consumption is not friendly to the environment. At present, the review of carbon footprint is becoming more and more strict, and it is not a good thing for enterprises to become ''big households'' in energy consumption. On the other hand, from a practical point of view, the cost of electricity is high, and the cost of maintaining a data center is huge: electricity costs account for a large percentage of current data center operating costs [8]. Therefore, it is critical to maintain energy efficiency in cloud data centers.

The quality of service (QoS) is another important metric for cloud providers [9], [10]. Generally, the QoS means that a network can utilize various basic technologies to provide better service capabilities for specified network communications. It is a security mechanism for the network and a technology for solving network delays and congestion. In this

paper we use QoS to assess the ability of a service provider to meet customer service needs.

Cloud users submit various computing tasks or resource reservations to cloud providers. Cloud providers provide services, e.g., infrastructure as a service (IaaS), platform as a service (PaaS), software as a service (SaaS) to generate revenue. A service level agreement (SLA) is a mutually agreed agreement between service providers and users to guarantee the performance and availability of services under certain overhead. Usually this overhead is a major factor to drive cloud providers to provide the QoS [8], [11], [12]. The optimal operating state of a service with a clear SLA is: adding additional resources to improve the system brings less revenue than the benefits of investing the resource in other services. When the cloud provider is failed to guarantee the SLA, he needs to compensate the cloud user for the loss. For example, in an Amazon S3 service level agreement, Amazon provides service commitment to users, which guarantees ''99.9% uptime per month''. The S3 SLA guarantees an average of 99.9% of all time slots in 5 minutes in a month. The most likely situation allowed by the SLA is equal to 40 minutes per month being unavailable. At the same time, service compensation is provided, that is, Amazon will provide service compensation if the SLA commitment is not met. If it does not meet the 99.9% service level, Amazon will reduce the 10% fee for the next month. If the availability drops below 99.0% and the conversion is equivalent to at least 7 hours of inability to serve within a month, Amazon will reduce the user's 25% fee.[1]

In order to guarantee high QoS, huge amounts of resources need to be provisioned in cloud data centers. However, managing and maintaining over-configured resources in turn leads to a large amount of energy costs, including the configuration and maintenance costs of cooling systems, physical components and other facilities [7], [13]–[15]. Therefore, one challenge that cloud service providers face is to ensure high service quality while controlling the energy consumption of data centers.

There are a lot of research works that have proposed many methods to optimize these two metrics based on various technologies, such as task scheduling [16], [17], VM migration [3], [18], server consolidation [19], [20], etc. In most of the above-mentioned literature, the authors use a population-based intelligent optimization algorithm, which often which often prematurely converges on high-dimensional complex problems and in large-scale data centers, the computational complexity of those algorithms can become extremely high. What's more, some of them are aimed at either saving energy or optimizing QoS [3], [16] and rarely take into account the hidden value of historical utilization data.

B. OUR CONTRIBUTIONS

Due to the proliferation of cloud users, cloud data centers need to deploy more servers to meet computing needs, which results in a large amount of carbon dioxide output and high operating costs. However, the users care about the quality of service provided, which drivers the service provider to provide reliable service while maintaining the data center operating cost within the controllable range. Therefore, the problem we need to solve is to find a scheme that can achieve a trade-off between energy consumption and QoS. The best compromise between cloud service quality and energy consumption can be achieved by blue virtualization technology [13], [21] and VM consolidation [19], [20], [22]–[24]. However, most consolidation methods are based on the current resource utilization of the host for immediate VM migration, without considering the future load state of the host.

In this paper, we propose an empirical forecast algorithm **(EFA)** to predict the load states of physical nodes in a dynamically changing heterogeneous cloud environment. The algorithm obtains the future state of a physical node by statistically analyzing the historical data and current utilization of the physical node. After the overloaded host is determined, the weight priority algorithm **(WPA)** assigns weights to several history factors of each migratable VM on the overloaded host, and the migration priority of each VM is calculated according to all weighting factors. The algorithm selects the VM with the highest priority for migration based on this priority list.

In summary, the main contributions of this article are as follows.

1) A two-level management model is proposed under the IaaS cloud environment.
2) An empirical forecast algorithm **(EFA)** based on historical utilization data of the host is proposed to predict the next possible state of the host, which determines whether the host is continuously overloaded by making full use of the potential value of historical information.
3) A weight priority algorithm **(WPA)** is proposed to determine the priority of migratable VMs on an overloaded host. This algorithm assigns weights to the several recent utilization factors of a VM.
4) The proposed approach is compared with the existing overloaded hosts detection algorithms with different VM selection policies under different workloads.

We enhance the energy efficiency of cloud data centers (minimizing energy consumption while maximizing the QoS). The high energy consumption results in much carbon dioxide (CO_2) emissions, which contributes to air pollution and global warming. Therefore, the research in this paper is of great significance to modern information technology and environmental protection. In addition to reducing the operational cost, the optimization of energy efficiency can also improve user satisfaction and promote the development of the cloud industry.

The rest of the paper is organized as follows. In Section II, the related work is introduced. In Section III, we describe the two-level management model and several metrics related to energy consumption and QoS. In Section IV, we describe the basic principles of the EFA algorithm and the WPA algorithm. In Section V, we introduce the specific parameter

[1] https://blog.csdn.net/chdhust/article/details/74086776

TABLE 1. Comparison of several research.

Schemes	Technique	Strategy	Disadvantage
Kusic, *et al.* [9]	Resource provisioning	Limited lookahead control (LLC)	High cost result by frequent switching of host status
Chen, *et al.* [37]	VM reallocation	Multi-objective optimization	VM selection strategy based on current CPU utilization
Melhem, *et al.* [38]	Server consolidation	Markov model	Invariant transition probability matrix
Yadav and Zhang [41]	Host detection	M-estimate regression	High time consumption

configuration used in the simulation experiment and analyze the experimental results. In Section VI, we summarize the paper and discuss the future research plan.

II. RELATED RESEARCH

A. IN TERMS OF ENERGY CONSUMPTION

During the past years, numerous researches have focused on saving energy in cloud center [3], [4], [16]–[19], [25]–[28]. Some of them reduced energy consumption by optimizing the task scheduling process [25], [27] and resource allocation issues [4], [16], [17], [26], [28]. In [16], a task scheduling method using a combination of cultural and ACO algorithm was presented in order to optimize energy consumption. However, the complexity of the combination algorithm is too high and they only consider the task makespan without evaluating the algorithm calculation overhead. In [26], a GreenSched model was proposed to schedule tasks on identified less energy consuming or energy aware nodes. Instead of turning off low-energy nodes, they use them as mission gathering places. Some of them improved the VM migration process to minimize energy consumption [3], [18], [19]. Tao *et al.* [18] reduced communication energy consumption by migrating VM in groups. In [3], Kliazovich *et al.* underlined the role of communication fabric in data center energy consumption. However, these papers simply discuss the energy consumption of the data center, they ignore the most important part of the data center, namely the QoS.

B. IN TERMS OF THE QoS

As a common performance requirement between cloud providers and cloud users, the QoS has always been the focus of industry and academia [29]–[31]. In [29], Liu *et al.* built a cost prediction model to predict virtual machine migration costs at the bottom level of migration. In [30], Deshpande *et al.* proposed GMGD to eliminate the retransmission of duplicate memory pages. However, gang migration can generate a lot of network traffic. The key issue studied by Zhang *et al.* [31] was how to arrange tasks on a hybrid cloud so as to maximize the profit of the cloud provider while satisfying the QoS requirements. They used a variant of the PSO algorithm and three hybrid algorithms to solve this problem. Service guarantees on hybrid clouds are much more complicated than single cloud. More research on data center performance optimization problems can be found in [12] and [32]–[34].

C. IN TERMS OF ENERGY CONSUMPTION AND QoS

Energy consumption and service quality are two highly relevant metrics in the cloud data center and many methods are proposed to simultaneously optimize them [**?**], [7], [9], [10], [23], [35]–[42]. In [9], Kusic *et al.* implemented and validated a dynamic resource allocation framework for a virtualized server environment. The authors controlled the power consumption and performance management of the virtualized computing environment through limited lookahead control (LLC). However, in their work, the host state is frequently switched, which easily damages the hardware. In [37], a multi-objective optimization management method with TOPSIS was introduced to achieve lower SLA violation rate, less energy consumption and better balance among different objectives. Melhem *et al.* [38] proposed a Markov perdiction model to avoid immediate VMs migration. They proved their superiority in terms of energy consumption and SLA violations. Nevertheless, in their work, the authors assumed that the transition probability between states is constant, which is not consistent with dynamic cloud environments. In [40] and [41], Yadav and Zhang introduced the M-estimate regression algorithm to calculate the upper CPU utilization threshold. However, the time complexity of their algorithm is high and the regression-based estimation model is not robust, so it is highly sensitive to outliers.

As shown in Table 1, the main disadvantage of existing works in literature is presented in the data center (e.g., [9], [37], [38], [41]). Our strategies not only consider the current host state, but also predict the future state of the host through historical data. We comprehensively evaluate multiple factors to determine when and how to migrate virtual machines.

III. SYSTEM MODEL

We consider a data center consisting of n heterogeneous physical nodes. Each physical node hosts multiple virtual machines. Each physical node H_i and virtual machine V_j^i (representing the jth virtual machine on physical node i) have these characteristics: CPU performance measured in millions of instructions per second (MIPS), certain RAM capacity and network bandwidth. Since systems usually share storage between communication servers through network attached storage (NAS) [29], [43] or storage area network (SAN) [44], [45], we do not need to know the application load and the time of VM configuration. Multiple independent users submit requests to configure m heterogeneous virtual

TABLE 2. Notations.

Symbol	Meaning
n	number of servers in the data center
m	number of VMs in the data center
H_i	the i-th server in the data center
V_j^i	the j-th virtual machine on physical node i
C_r^i	the requested CPU capacity to the physical node i
C_c^i	the total CPU capacity of the physical node i
λ	the number of weighting factors of WPA algorithm
EC_i^{full}	maximum energy consumption when the server i is fully utilized
ρ	the fraction of energy consumed by a idle server
EC_i^{idle}	energy consumption when the server i is idle
$U_i(t)$	CPU utilization of the server i at time t
$EC_i(t)$	the energy consumption of the i-th server at time t
EC_i	The total energy consumption of the server i
x_i	flag variable to mark whether the server i is shut down
T_i^a	the total time that the server i is active
T_i^f	the total time that the server i experienced 100% utilization
C_j^r	the total required CPU capacity during the lifetime of the virtual machine j
C_j^d	performance degradation due to the migration of the virtual machine j
R_j	the RAM capacity of virtual machine j
B_a	available bandwidth for VM migration
T_m^j	the migration time of the virtual machine j
C_j	CPU capacity of the virtual machine j (in MIPS)
p_{v_j}	migration priority of virtual machine j

machines via the Internet. The user establishes a service level agreement (SLA) with the cloud provider. A SLA violation occurs when the total CPU performance requirement C_r^i allocated to the physical node H_i exceeds the total CPU capacity C_c^i of the node.

Table 2 enumerates all the symbols used in the paper.

We present the target system architecture in Figure 1. Our model consists of two parts: **the global manager** and **the host manager**. The host manager resides on each physical node, which downwards communicates with the VMM and upwards interacts with the global manager. The global manager resides on the master node. It obtains information from the host manager and issues optimization commands based on the collected information. The **VMM** performs VM migration and changes the host's mode. It is the master agent that monitors host load activity.

The host manager contains the following modules.

- **Log module:** This module monitors and records the host load, which is equivalent to the VMM subagent.
- **Host detection module:** This module predicts the future state of the host based on the data from the log module.
- **VM selection module:** This module determines the VM to be migrated according to a specific strategy.

The global manager contains the following modules.

- **Database:** This module stores related information of all hosts, including the metadata and resource usage of hosts.
- **Optimization module:** This module triggers the VM migration process and issues commands that optimize VMs allocation.

A. POWER MODEL

The power consumption of computing nodes in cloud data centers is mainly determined by the CPU, memory, disk storage and network interfaces. In this paper, we mainly study the energy consumption generated by the CPU. One of the most common energy consumption models is the linear model, where there is a linear correlation between power consumption and CPU utilization [2], [3], [38], [46]. and these studies [2], [6], [7], [47] demonstrate that a idle server consumes 70% of the full-load power. Therefore, in our target system, we define the energy consumption of the i-th server at time t as

$$EC_i(t) = \begin{cases} \rho \times EC_i^{full} & \text{the } H_i \text{ is idle,} \\ EC_i^{idle} + (1-\rho) \times EC_i^{full} \times U_i(t) & \text{the } H_i \text{ is busy.} \end{cases}$$

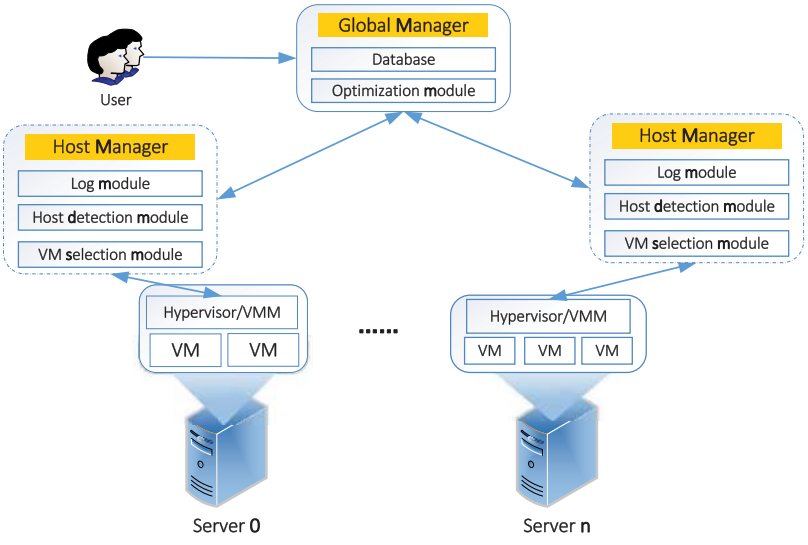

FIGURE 1. Two-level management model.

TABLE 3. The electric energy consumption at different workloads.

Server name	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
HP ProLiant ML110 G4	86	89.4	92.6	96	99.5	102	106	108	112	114	117
HP ProLiant ML110 G5	93.7	97	101	105	110	116	121	125	129	133	135

When the server is idle, ρ is a static coefficient representing the energy ratio of the idle processor (i.e., 70%). The EC_i^{full} represents the energy consumption of the physical node i under full load. Since the CPU utilization dynamically changes according to the workload, the CPU utilization is a function of the time t as an independent variable, which is denoted as $U(t)$. In our experiment, the EC_i^{full} is set to $250W$ as used in [35]. The energy consumption of the servers used in our experiment is obtained from the SPECpower.[2] The electric energy consumed by the considered servers at different workloads is shown in Table 3. The energy consumption of server H_i throughout the process can be expressed as

$$EC_i = \int_{t_0}^{t_1} EC_i(t)dt.$$

Then, the total energy consumption of a cloud data center with n nodes is

$$EC = \sum_{i=1}^{n} x_i EC_i, \tag{1}$$

where

$$x_i = \begin{cases} 0, & \text{the } H_i \text{ is shutdown,} \\ 1, & \text{other.} \end{cases}$$

B. SLA VIOLATION METRICS

In a cloud data center, the SLA is formal representation of the QoS that cloud providers provide to cloud users. A SLA violation occurs when users submit an excess demand to the data center. If the host oversubscribes, its computing performance will be greatly reduced. Similarly, the virtual machine migration process consumes additional resources. The degree of SLA violation directly reflects the availability and robustness of the system. We select several metrics defined in [38] to measure the degree of SLA violation.

1) **SLATPAH:** the percentage of time each active host violated the SLA, which is expressed as

$$SLATPAH = \frac{1}{n}\sum_{i=1}^{n}\frac{T_i^f}{T_i^a}, \tag{2}$$

where n is the number of servers; T_i^a represents the total time that host i is active; T_i^f represents the total time that host i experiences full load.

2) **SLAPDM:** the performance degradation caused by migration, which is expressed as

$$SLAPDM = \frac{1}{m}\sum_{j=1}^{m}\frac{C_j^d}{C_j^r} = \frac{1}{m}\sum_{j=1}^{m}\frac{0.1 \times C_j}{C_j^r}, \tag{3}$$

where m represents the number of VMs in the data center; C_j^r represents the total required CPU capacity (in MIPS) during the life cycle of virtual machine j; C_j^d represents a decrease in performance due to virtual machine j migration. It is generally believed that the virtual machine migration will lose 10% of the CPU computing power [38]. The C_j represents the CPU power (in MIPS) of virtual machine j.

Therefore, the overall SLA violation is defined as the product of (2) and (3), i.e.,

$$SLAV = SLAPAH \times SLAPDM. \tag{4}$$

Considering the overall performance, energy consumption, SLA violations and other metrics, we use the performance metric **Pertric** in [41] to measure the efficiency of the algorithm, which is expressed as

$$Pertric = EC \times SLAV \times NHS, \tag{5}$$

where the NHS represents the number of shutdown hosts.

IV. EMPIRICAL OPTIMIZATION ALGORITHM

The main purpose of this study is to reduce energy consumption and guarantee QoS. The goal of reducing energy consumption can be achieved by switching the idle node to a low energy consumption mode [35], [48]. According to the current resource requirements of VMs, the VM migration can be used to dynamically consolidate the VMs onto a minimum number of physical nodes. Dynamic consolidation can be divided into three parts:

1) **Host overload/underload detection:** In the data center, the load state of a physical node is related to the energy consumption metric of the system. An overloaded host will affect the response time and the QoS, while an underloaded host will lead to more energy consumption. Therefore, the first thing is to detect if the host is overloaded or underloaded.
2) **VM selection:** After determining which host is in an overloaded state, one or more VMs need to be migrated from the host, or a host is in a underloaded state, it is necessary to migrate all VMs from the host and switch it to sleep mode to reduce power consumption.
3) **VM placement:** A new candidate host set is found to place migrated VMs. It should be sure that the newly added VM does not cause the new placement point to be overloaded.

As illustrated in Figure 2, the detailed process can be described as follows: (1) The log module on the host manager collects utilization data of each VM and sends all the data to the database of the global manager. (2) In the host

[2]http://www.spec.org/power_ssj2008/results/

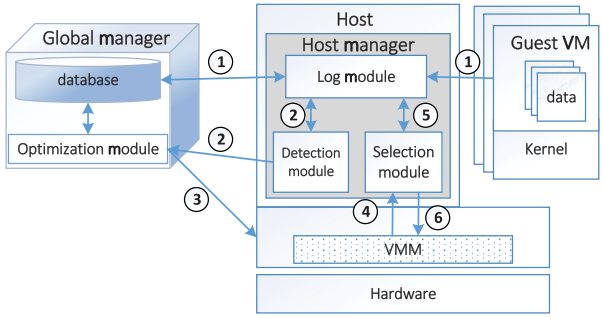

FIGURE 2. System architecture.

detection module, the detection algorithm predicts the host state by analyzing the data of the log module. The detection module feeds back the results to the optimization module of the global manager. (3) The optimization module issues a migration command and an optimization placement command to the VMM of the host node according to the detection results and the data from the database. (4) The VMM triggers the migration activity and enables the selection module. (5) In the selection module, the selection algorithm selects the VM to migrate based on the data in the log. (6) The VMM receives the migrated virtual machine and completes the VM placement.

Data collection is the preparation phase of the consolidation process. We mainly take optimization measures in Stage (2) and Stage (5). Next we discuss these two sub-problems.

A. HOST DETECTION ALGORITHM

The previous works [35], [49] judge whether the host is overloaded according to the current host utilization. The VM begins to migrate immediately as soon as the host is overloaded. It is failed to consider that the load state at this time may be an illusion that the host is overloaded temporarily, which leads to unnecessary migration and increases power loss and performance degradation. The previously proposed dual-threshold scheme THR [35] obtains the host performance in a static manner. The static threshold [36], [38] is not suitable for the dynamic environment. When the workload in the cloud data center fluctuates slightly, a high utilization threshold can be set to use effectively resources. Once there is a surge in workload, the threshold will be adjusted to prevent the cloud from exploding due to the high server workloads, which will damage the service life of the physical node hardware facilities. In our two-level management system, the utilization threshold for each heterogeneous host is different and dynamic.

In this paper, we propose an empirical forecast algorithm **(EFA)**. The algorithm uses the historical data of a host to analyze the its next possible load state. There are only three load states for an active host: underloaded (“U”), normal (“N”) and overloaded (“O”). If the current state of the host is overloaded, we must first determine if its next state remains overloaded. The load may be self-adjusting due to the

Algorithm 1 Empirical Forecast Algorithm (EFA)

Input: the host H_i
Output: *isOverloaded*

function isHostOverUtilized(U_{H_i})
  isOverLoaded $\leftarrow$ *false*
  get the utilization list U_{H_i} of the H_i
  upThreshold $\leftarrow 1 - \alpha \times mad(U_{H_i})$
  $S_{H_i} \leftarrow getStateHistory(U_{H_i}, upThreshold)$
  state $\leftarrow$ *null*
  calculate the $\overline{X_{H_i}}$ according S_{H_i} with the Equation (8)
  if the probability elements in $\overline{X_{H_i}}$ are different **then**
    get the *state* with the *maxProbability*
  else
    calculate the frequency of elements with same probability
    get the *state* with the *maxFrequency*
  end if
  if *state.equals*(*O*) **then**
    isOverloaded $\leftarrow$ *true*
  end if
  return *isOverloaded*
end function

dynamic nature of the system. If a host remains overloaded for a period of time, it indicates the necessity of VM migration on the host.

In EFA algorithm, the CPU utilization threshold is given as the median absolute deviation (MAD) [7]. Since the utilization record time is changing, the resulting median absolute deviation is also dynamically changed. MAD is a robust measure of sample difference in a univariate data set. MAD is a robust statistic that handles outliers in the dataset is more elastic than the standard deviation, greatly reducing the impact of outliers on the dataset.

For a univariate data set $D_1, D_2, ..., Dn$, the MAD is calculated as:

$$MAD = median_i(|D_i - median_j(D_j)|). \tag{6}$$

The MAD is defined as the median of the absolute deviation of the data point to the median. We define the upper utilization threshold *upThreshold* as:

$$upThreshold = 1 - \alpha \times MAD, \tag{7}$$

where $\alpha \epsilon \mathbb{R}^+$ is used to adjust the extent to which the system consolidates the VMs. In our algorithm, we set the value of α to 1.0.

Given a host M and its utilization history $U_M = \{U_0, U_1, U_2, ...U_{l-1}\}$, where l is the length of the history record. We translate the utilization history into host states records in Algorithm 1. Supposing we get the corresponding host states list as $S_M = \{S_0, S_1, ...S_{l-1}\}$ and the current status of host M is S', the transition probability of state S_t at time t can be inferred from the Equation (8):

$$P(S_t|S') = \frac{P(S_t, S')}{P(S')}. \tag{8}$$

Then we can get a state transition vector $\overline{X_M} = \{X_0, X_1, X_2\}$ and each of these vector elements represents the transition probability of three states.

For example, given $S_M = \{U, U, N, N, O, N, U, O, N, O, O, O\}$ and the current state S' of the host M is "O". The probability that the next state of M is "O" is calculated as

$$P(O|S') = \frac{P(S_t = O, S' = O)}{P(S' = O)} = \frac{2}{5} = 0.4,$$

where $P(S' = O) = P(S_t = O, S' = O) + P(S_t = N, S' = O) + P(S_t = U, S' = O)$. Similarly, we have $P(N|S' = O) = 0.4$ and $P(U|S' = O) = 0.2$. Therefore, the state transition vector $\overline{X_M}$ is $\{0.4, 0.4, 0.2\}$. According to the probability vector, we can know that the next possible state of the host M is "O" or "N". In this step, we may encounter two situations:

1) There are three different probabilities in the probability map. The state with the largest transition probability is the target state.
2) There are two or more identical transition probabilities in probability map. We further calculate the frequency of states with the same probability in the recent half of S_M. That is, the number of occurrences of the same probability state in $S'_M = \{S_{l-\frac{l}{2}}, S_{l-\frac{l}{2}+1}, ..., S_{l-1}\}$ is calculated. Therefore, we only need to calculate the frequency f_O of the overload state "O" and the frequency f_N of the normal state "N" in S'_M. We get $f_O = 4$ and $f_N = 1$, which indicates that the host M remains continuously overloaded in the near future, at which point the VM migration is enabled.

After determining the overloaded host, the next step is to select the VM to be migrated on the overloaded host.

B. VM SELECTION ALGORITHM

It is critical to choose the VM on the overloaded host for migration. Improper selection results in that the host still unable to get rid of the overload state after migrating a VM. Then additional virtual machines need to migrate, which not only affects the QoS of the host, but also results in overall performance degradation.

We propose a weighted priority selection algorithm **(WPA)** to select VMs for migration in order to reduce the number of migrations as much as possible. The corresponding pseudo code is given in Algorithm 2. Our strategy is to set the priority for each migratable VM of the overloaded host and determine the order of the migration of the virtual machines according to the priority. The priority of the VM is calculated by Equation (9).

In the experiment of Section V, we adjust the value of λ several times to obtain the best number of weighting items. For each VM in the *migratableVm* list, we consider the recent utilization of λ terms $\{u_0, u_1, ..., u_{\lambda-1}\}$ and the reciprocal of the migration time $\frac{1}{T_m^j}$. The less the migration time is, the smaller the performance degradation is [30], [50]. Therefore, we set the maximum weight $\lambda + 1$ for $\frac{1}{T_m^j}$ and the closer

Algorithm 2 Weighting Priority Algorithm (WPA)

Input: the list *migratableVms*, λ
Output: *vmToMigration*

```
vmToMigration ← null
for VM V_j in migratableVms do
    p_{v_j} ← 1
    get the migration time T_m^j of V_j
    W_{T_m^j} ← λ + 1
    p_{v_j} ← p_{v_j} + W_{T_m^j} × T_m^j
    for t = 0 to (λ − 1) do
        W_{u_t^j} ← λ − t
        p_{v_j} ← p_{v_j} + W_{u_t^j} × u_t^j
    end for
    p_{v_j} ← p_{v_j} / (λ + 1)
end for
get the the max priority p_{v_m} in migratableVms
vmToMigration ← V_m
return vmToMigration
```

the utilization rate is to the current time, the greater the weight value is. Given the equation to calculate the VM priority,

$$p_{v_j} = \frac{\sum_{t=0}^{\lambda-1} W_{u_t^j} \times u_t^j + (T_m^j)^{-1} \times W_{T_m^j}}{\lambda + 1}, \tag{9}$$

where the migration time is calculated as $T_m^j = \frac{R_j}{B_a}$, the $W_{u_t^j} = \lambda - t$ represents the weight of utilization factor u_t^j and u_t^j represents the utilization rate of virtual machine j at time t.

C. VM PLACEMENT STRATEGY

Virtual machine placement issue [14], [18], [32], [38], [46], [51]–[54] can be seen as a variable-size bin-packing problem. The heterogeneous hosts represent different bins and the number of hosts is the number of bins. The size of the bin refers to the available CPU capacity of the host, and the price corresponds to the energy consumption of the node. Since the packing problem is a typical NP-hard problem [55], in order to solve the VM placement problem, we adopt the PABFD algorithm used in [32] and [35]. The algorithm idea is to sort all VMs in descending order according to the current CPU utilization of the VM and place the VM which results in the least increase in energy consumption on the target host. This algorithm is an optimized version of the Best Fit Decreased Algorithm (BFD) [13].

V. PERFORMANCE EVALUATION

In this section we describe the experimental setup and experimental results. We chose CloudSim [56] cloud computing simulation platform for algorithm performance testing, which provides the following features: (1) a tool supporting the modeling and simulation of large-scale cloud computing infrastructure; (2) a self-contained platform supporting data centers, service agents, scheduling, and allocation strategies.

Therefore, the ClouSim is very suitable for cloud computing simulation experiments.

A. SIMULATION SETUP

To ensure the validity and comparability of the experiment, we use the experimental configuration as same as in [7]. We simulate the algorithms under two kinds of workloads. The PlanetLab data is provided as a part of the CoMon project.[3] It is a monitoring infrastructure for PlanetLab. In this project, CPU utilization data is obtained from more than a thousand VMs of servers at five-minute intervals, and these servers are located in more than 500 locations around the world. The data is stored in ten different files. We selected two days from the workload tracking. Through simulation, each VM randomly distributes workload tracking from one VM on the corresponding date. In the Random workload, each VM runs an application with the variable workload, which is modeled to generate the utilization of CPU according to a uniformly distributed random variable. Actually, whenever a random function generates a set of workload automatically as a cloudlet we called Random workload.

TABLE 4. Hosts instance types.

Host instance tyeps	Hosts_Mips	Host_Pes	Host_Ram
HP ProLiant ML110 G4	1860	2	4GB
HP ProLiant ML110 G5	2660	2	4GB

TABLE 5. VMs instance types.

VM instance types	VM_Mips	VM_Ram(GB)
High-CPU Medium Instance	2500	0.85
Extra Large Instance	2000	3.75
Small Instance	1000	1.70
Micro Instance	500	0.61

Then two types of heterogeneous physical hosts are defined (see Table 4). Both of these hosts are dual-core CPUs with computing power of 1860 Mips and 2660 Mips respectively. Because the workload dataset provided by the PlanetLab is a bit old, there are some following reasons for choosing the dual-core CPU: if we take more core CPU, VM consolidation will not work properly. For example, if we take like 128 core CPU servers, the change of server overloading is very low. Therefore, we can't calculate SLA violation and VM migration in the proposed method. Four heterogeneous VM types are defined according to the Amazon EC2 instance types[4] (see Table 5). The four VM instance types are single-core because the PlanetLab workload data used for simulation comes from a single-core VM.

B. λ-VALUE ADJUSTMENT

Since we propose the λ term factors weighting operation in the WPA algorithm, λ is the most important parameter of the WPA algorithm. In this section, we perform an adjustment test on the weighted term λ of the WPA algorithm to obtain the optimal value of λ. In the experiment, we set the total history length of the VM utilization list to 30. We test the effects of different λ-values on the WPA algorithm. We have selected a range for λ-value from 1 to 20. We have not increased the range over 20 because of time complexity.

In order to obtain the best λ value, we rely on several of the aforementioned metrics. In addition, we also refer to two additional metrics: NVMM and NHS. The NVMM is the number of VM migration that occur in the data center during the entire simulation. The NHS is the total number of host shut down in the data center. Since the experimental results show that the WPA algorithm has similar trends under random and real workloads, we only describe the results of random workload as a representative. The impact of λ on various metrics is illustrated in Figure 3 and Figure 4.

According to these figures, we find that all metrics are at the best level when λ is 4. From the perspective of energy consumption and SLA violations, low energy consumption come at the expense of SLA (see Figure 3(a) and (f)). According to Equation (4), there is no doubt that changes in SLATPAH and SLAPDM are directly affect the SLAV metric (see Figure 3(b), (c) and (f)).

We show the trends of the WPA algorithm with different λ values on the **Pertric** metric in Figure 4. The result shows that when the λ-value is 4, the algorithm works best. Therefore, in the following experiment, we defined λ in the WPA algorithm as 4.

C. EXISTING ALGORITHMS

Many algorithms have been proposed to improve the energy efficiency of cloud data centers (minimizing energy consumption while maximizing service quality). The existing algorithms are described as follows.

1) **MeMs** [41]: This algorithm creates an upper CPU utilization threshold using M-estimate regression to detect overloaded hosts and then combines the MuMs [40] VM selection scheme to optimize VM consolidation.
2) **LrMmt** [7]: The main idea of LrMmt is fitting simple models to localized subsets of data to build up a curve that approximates the original data with considering the migration time of the VM.
3) **MadMu** [7]: The main idea of MadMu is to adjust the value of the upper utilization threshold depending on the strength of the deviation of the CPU utilization. In the VM selection phase, the algorithm selects the VM with the lowest CPU utilization for migration. After multiple iterations, the host utilization is below the upper threshold.
4) **IqrMc** [7]: The IQR which is called the midspread or middle fifty, is a measure of statistical dispersion, being equal to the difference between the third and first quartiles. The IQR is a robust statistic, having a breakdown point of 25% and is thus often preferred

[3]https://www.planet-lab.org/planetlablogs
[4]https://aws.amazon.com/cn/ec2/instance-types/

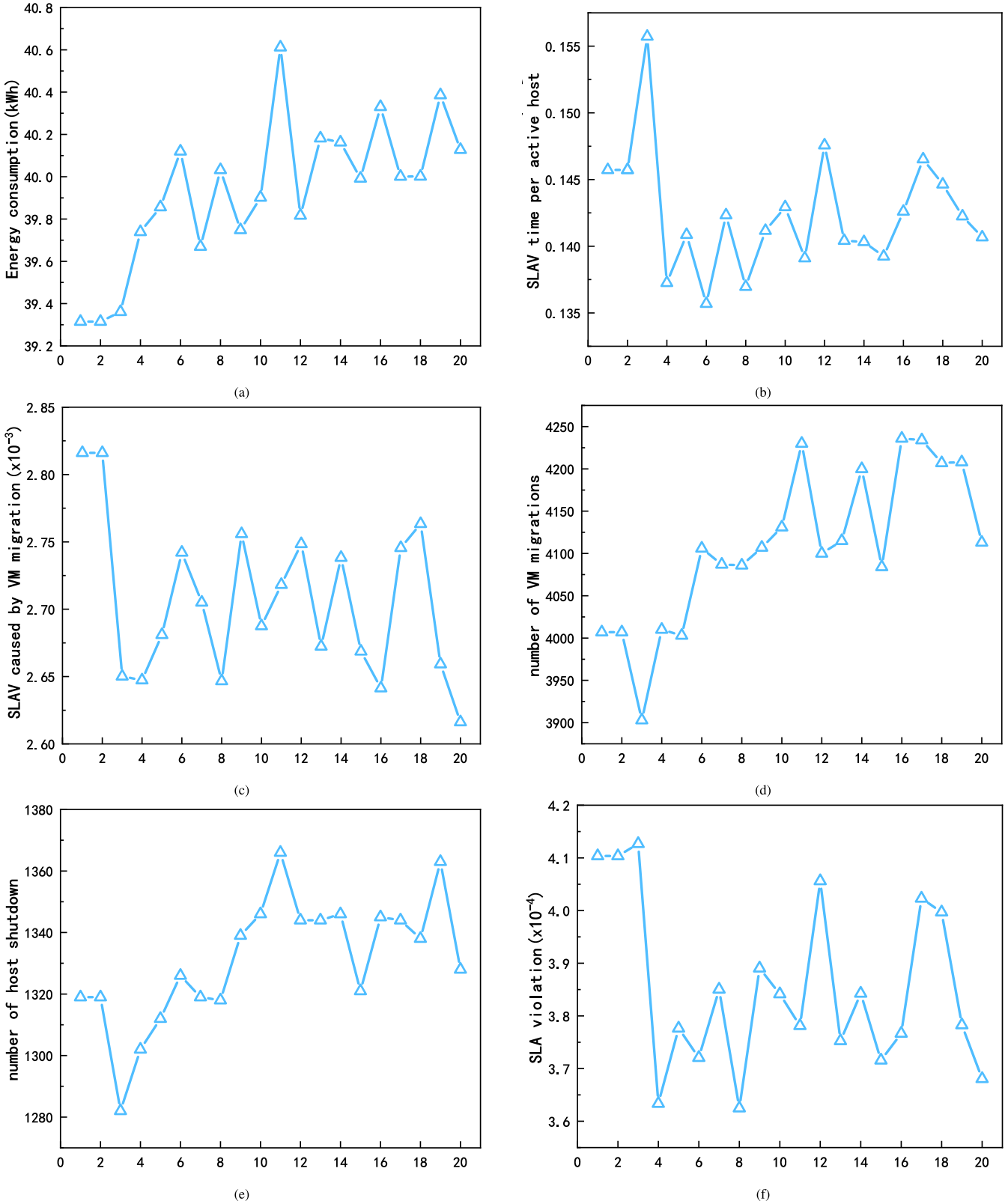

FIGURE 3. The impact of λ parameters on WPA performance (illustrated by the case of random workload). (a) The EC metric. (b) The SLATPAH metric. (c) The SLAPDM metric. (d) The NVMM metric. (e) The NHS metric. (f) The SLAV metric.

to the total range. The IqrMc selects those VMs to be migrated that have the highest correlation of the CPU utilization with other VMs.

We refer to the method in this paper as EfWp, which combines the EFA host overload detection algorithm with the WPA VM selection algorithm. Since the superiority of

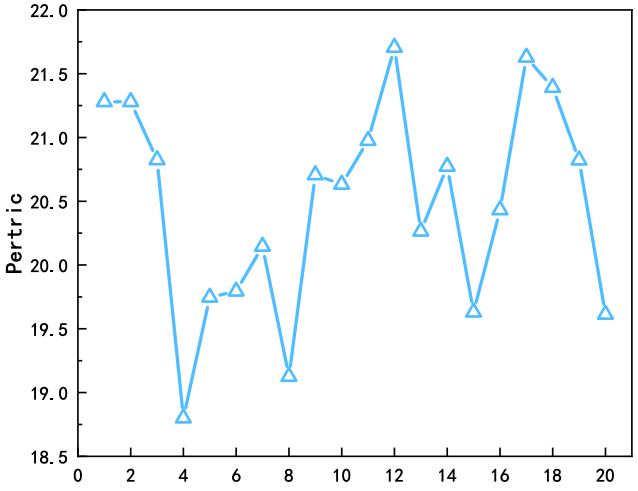

FIGURE 4. The impact of λ on Pertric of the WPA (illustrated by the case of random workload).

the LrMmt algorithm compared with other traditional algorithms has been proved in [7], we compare the LrMmt and MeMs [40], [41] with the EfWp algorithm. In order to get a more intuitive conclusion, we also select two other algorithms, MadMu and IqrMc [7], as reference algorithms.

D. EXPERIMENTAL RESULTS AND ANALYSIS

In this experiment, we simulate the EfWp algorithm and compare it with the existing algorithms described in V-C.

As illustrated in Figure 5 and Figure 6, the EfWp method has achieved significant results in terms of energy consumption, SLA violations, NHS and NVMM and energy efficiency. These algorithms have similar trends in performance, no matter the random workload or the real workload, indicating that the EfWp algorithm is highly robust and feasible under variable workloads. In the following sections, we will discuss the energy consumption, SLA violations, number of host shutdowns, number of VM migrations, and energy efficiency for these algorithms.

1) ENERGY CONSUMPTION

For the five algorithms (EfWp, MeMs, LrMmt, MadMu, and IqrMc) with different workloads, the energy consumption is shown in Figure 5(a), which shows the energy consumption for the five algorithms. The EfWp is slightly better than the MeMs algorithm (1.62% in random workload and 2.23% in PlanetLab workload, respectively). The reason is that the MeMs may ignore interaction effects and nonlinear causality while our EfWp algorithm considers the impact of recent utilization records on host future status. In addition, the EfWp algorithm is also superior to the LrMmt algorithm (11.40% in random workload and 26% in PlanetLab workload, respectively) and far better than MadMu algorithm (34.36% in random workload and 39.23% in PlanetLab workload, respectively) and IqrMc algorithm (33.82% in random workload and 32.19% in PlanetLab workload, respectively) in terms of energy comsumption. This can be explained by the fact that the VM selection strategy in EfWp is based on the minimum migration time and also considers the recent VM requirements. The impact factor is maximized and weighted to achieve accurate VM selection. Therefore, EfWp can achieve more friendly energy consumption due to its simple calculation compared to other algorithms.

2) OVERALL SLA VIOLATIONS

In terms of overall SLA violations (see Figure 5(b)), The MeMs have the highest SLA violation rate because the algorithm focuses on reducing energy consumption, with the improvement of service quality as an auxiliary effect. The EfWp has the lowest SLA violation rate. It is far superior to the MeMs algorithm (79.33% in random workload). Although the other three algorithms (LrMmt, MadMu and IqrMc) achieve low SLA violations, EfWp is still slightly better than them both in random and PlanetLab workload. The EfWp and MadMu use the same strategy to dynamically obtain the upper threshold of host utilization. The former additionally considers the recent use of host computing resources, and the latter simply compares the current computing demand with the upper threshold. Under PlanetLab workload, the SLA violation rate of EfWp and MeMs are not the same order of magnitude. As can be seen from Section III-B, the overall SLA violation is closely related to the violation rate of each host and the SLA violation caused by VM migration.

3) NUMBER OF HOST SHUTDOWNS AND VM MIGRATIONS

At the same time, EfWp's significant improvement in the number of host shutdowns (NHS) and migrating VMs (NVMM) has established its superiority in SLA violations which is illustrated in Figure 5(c) and (d). The NHS and NVMM are directly proportional to performance degradation. Frequent host state switching and virtual machine migration have a great impact on the stability of the entire system. The EFA host detection algorithm predicts the host state at the next moment based on the recent utilization rate, which can more accurately capture the load of the host, thereby avoiding the restart of the shutdown host. That is, the algorithm reduces redundant host reactivation. Figure 5(c) shows that MadMu has the highest number of host shutdowns (3132 in random workload and 6274 in PlanetLab workload), and the number is far greater than the total number of hosts (100 in random workload and 800 in PlanetLab workload) indicating that many hosts are reactivated after they are shut down, which runs counter to our goal. The strategy adopted by IqrMc is similar to that of MadMu, which explains the performance of IqrMc on the NHS metric. The NHS and NVMM metrics have a positive correlation. When all VMs on the host are migrated, the host can be shut down, therefore the more VM migrations, the more frequently the host reactivates. This is also the reason why Figure 5(d) has a curve similar to that of Figure 5(c).

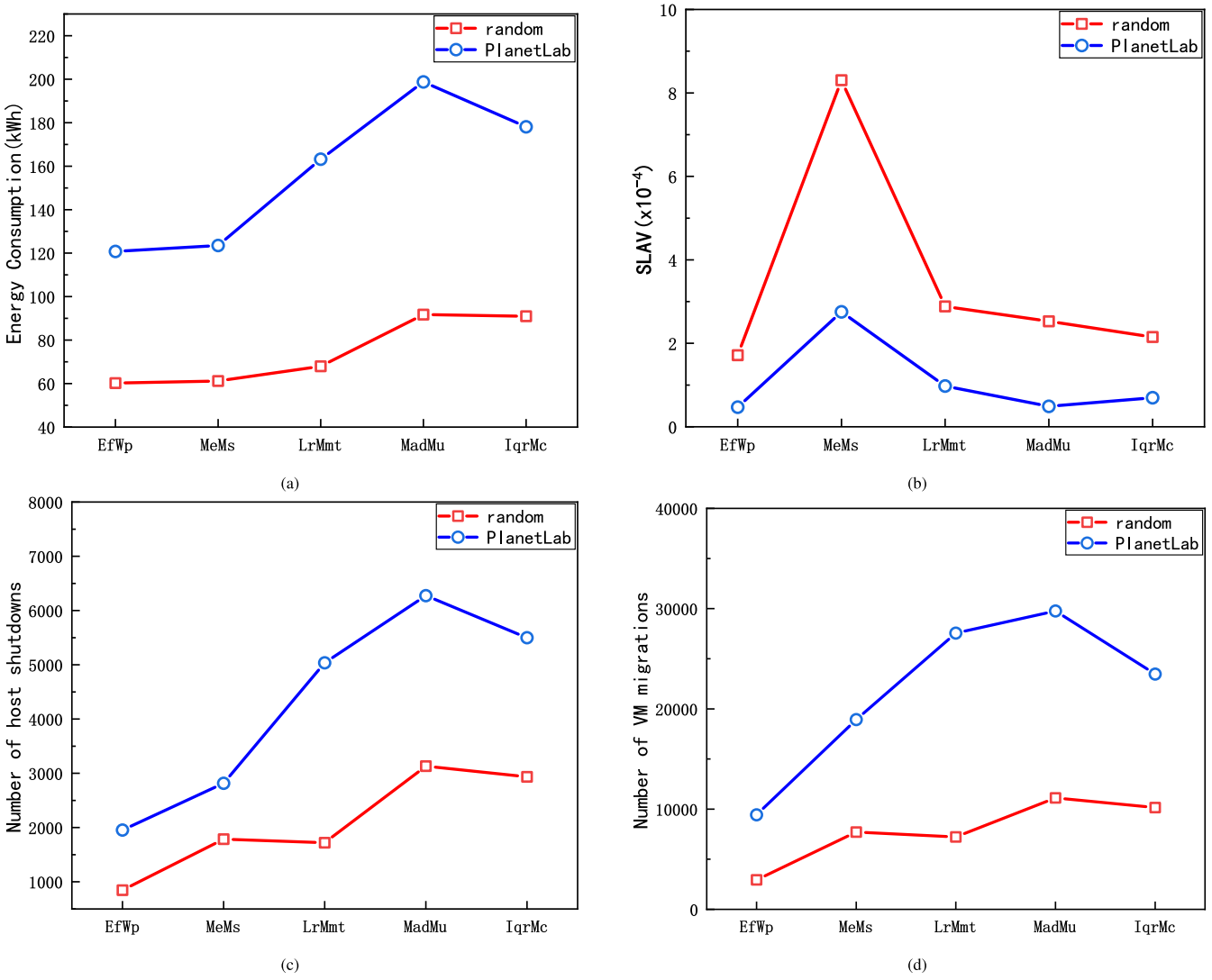

FIGURE 5. The comparison of the simulation results of metrics (i.e. enegry consumption, SLA violation, number of shutdown host and number of VM migration). (a) The energy consumption. (b) The SLA violation. (c) The number of shutdown hosts. (d) The number of VM migration.

4) PERTRIC

We use the overall performance metric **Pertric** proposed in [41] to comprehensively evaluate our algorithm. The **Pertric** metric analyzes all energy-related aspects of the cloud data center, such as minimizing power loss, overall SLA violation rates and the number of hosts that have experienced state transitions during load balancing. Figure 6 shows the effectiveness of the EfWp over other existing algorithms. The EfWp has the best energy efficiency metric among the five algorithms (EfWp, MeMs, LrMmt, MadMu, and IqrMc) which is due to the simplicity and effectiveness of its strategy (see Figure 5). The MeMs have the highest Pertric, while our goal is to minimize the Pertric. Because the high violation rate (Figure 5(b)) of the MeMs neutralizes its efforts in energy consumption, the overall performance of MeMs is not as good as our algorithm EfWp, so do other three algorithms. Further, we conduct a joint hypothesis test on the **Pertric** metric to analyze the trade-off between energy consumption and QoS for our proposed approach. As illustrated in Table 6, the F ratio (6.58) is greater than the F critical value (5.19), which indicates that the null hypothesis is rejected and the population means are significantly different from one another at the 0.05 level with p value of 0.031569 ($p < 0.05$). Therefore, the EfWp algorithm is significantly different from MeMs, LrMmt, MadMu and IqrMc.

5) TIME METRIC

Since computation overhead is an important metric to evaluate the algorithm, one sample t-test of VM migration time duration and host running time is also carried out. The average value of the sample mean times before a VM migration during the host detection underload or overload is 19.89 seconds with a 95% CI:19.6,20.18.The average value of the sample means host running time before transition to energy-saving-mode is 31.89 minutes with 95% CI: 31.9,36.46.

TABLE 6. The one-way ANOVA test in term of Pertric.

Source of variation	SS	df	MS	F	P-value	F crit
Between group	6186.708	4	1546.677	6.580778	0.031569	5.192168
Within groups	1175.148	5	235.0295			
Total	7361.856	9				

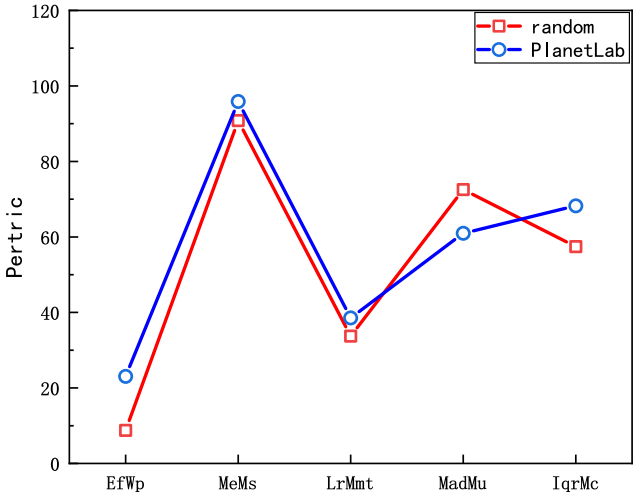

FIGURE 6. The comparison of the simulation result of the Pertric metric.

VI. CONCLUSION

In the data center, the balance between energy consumption and QoS is a win-win situation for service providers and cloud users. The original intention of our research is to seek the best compromise between energy consumption and service quality. For this purpose, we propose a two-level management model under a heterogeneous cloud environment. In this model, we propose an empirical forecast host detection algorithm (**EFA**) and a weighted priority VM selection algorithm (**WPA**) based on historical data. The **EFA** algorithm has the characteristics similar to the Markov chain. The difference is that the host state transition probability is updated in real time, which is more in line with the dynamic nature of the cloud data center. The **WPA** algorithm not only considers the migration time but also considers the recent utilization level.

As part of the future, we plan to further extend this work by studying network traffic-aware VM migration, considering the network communication overhead between related VMs. The implementation of these algorithms in the open source real cloud platform such as OpenStack would also be studied.

REFERENCES

[1] T. Rings *et al.*, "Grid and cloud computing: Opportunities for integration with the next generation network," *J. Grid Comput.*, vol. 7, no. 3, pp. 375–393, 2009.

[2] R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang, and X. Zhu, "No power struggles: Coordinated multi-level power management for the data center," *Solutions*, vol. 36, no. 1, pp. 48–59, 2008. [Online]. Available: http://portal.acm.org/citation.cfm?id=1346289

[3] D. Kliazovich, P. Bouvry, and S. U. Khan, "DENS: Data center energy-efficient network-aware scheduling," *Cluster Comput.*, vol. 16, no. 1, pp. 65–75, 2013.

[4] F. Juarez, J. Ejarque, and R. M. Badia, "Dynamic energy-aware scheduling for parallel task-based application in cloud computing," *Future Gener. Comput. Syst.*, vol. 78, pp. 257–271, Jan. 2018. doi: 10.1016/j.future.2016.06.029.

[5] U. S. Energy Information Administration. (May 2016). *International Energy Outlook 2016*. [Online]. Available: www.eia.gov/forecasts/ieo/pdf/0484(2016).pdf

[6] B. Heller *et al.*, "ElasticTree: Saving energy in data center networks," in *Proc. 7th USENIX Conf. Netw. Syst. Design Implement.*, 2010, pp. 249–264. [Online]. Available: http://dl.acm.org/citation.cfm?id=1855711.1855728TOKEN=48545471

[7] L. Adhianto *et al.*, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers," *Concurrency Comput. Pract. Exper.*, vol. 22, no. 6, pp. 685–701, 2010.

[8] Z. Cao and S. Dong, "An energy-aware heuristic framework for virtual machine consolidation in Cloud computing," *J. Supercomput.*, vol. 69, no. 1, pp. 429–451, 2014.

[9] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy, and G. Jiang, "Power and performance management of virtualized computing environments via lookahead control," *Cluster Comput.*, vol. 12, no. 1, pp. 1–15, Mar. 2009.

[10] S. Xue, Y. Zhang, X. Xu, G. Xing, H. Xiang, and S. Ji, "*QET*: A QoS-based energy-aware task scheduling method in cloud environment," *Cluster Comput.*, vol. 20, no. 4, pp. 3199–3212, 2017.

[11] H. Yuan, J. Bi, M. Zhou, and K. Sedraoui, "WARM: Workload-aware multi-application task scheduling for revenue maximization in SDN-based cloud data center," *IEEE Access*, vol. 6, pp. 645–657, 2017. [Online]. Available: http://ieeexplore.ieee.org/document/8110607/

[12] Y. Zhang and J. Sun, "Novel efficient particle swarm optimization algorithms for solving QoS-demanded bag-of-tasks scheduling problems with profit maximization on hybrid clouds," *Concurrency Comput., Pract. Exper.*, vol. 29, no. 21, p. e4249, 2017. doi: 10.1002/cpe.4249.

[13] L. Luo, W. Wu, W. T. Tsai, D. Di, and F. Zhang, "Simulation of power consumption of cloud data centers," *Simul. Model. Pract. Theory*, vol. 39, pp. 152–171, Dec. 2013. doi: 10.1016/j.simpat.2013.08.004.

[14] W. Fang, X. Liang, S. Li, L. Chiaraviglio, and N. Xiong, "VMPlanner: Optimizing virtual machine placement and traffic flow routing to reduce network power costs in cloud data centers," *Comput. Netw.*, vol. 57, no. 1, pp. 179–196, 2013. doi: 10.1016/j.comnet.2012.09.008.

[15] F.-H. Tseng, X. Chen, L.-D. Chou, H.-C. Chao, and S. Chen, "Support vector machine approach for virtual machine migration in cloud data center," *Multimedia Tools Appl.*, vol. 74, no. 10, pp. 3419–3440, 2015. [Online]. Available: http://link.springer.com/10.1007/s11042-014-2086-z

[16] P. Azad and N. J. Navimipour, "An energy-aware task scheduling in the cloud computing using a hybrid cultural and ant colony optimization algorithm," *Int. J. Cloud Appl. Comput.*, vol. 7, no. 4, pp. 20–40, 2017.

[17] N. Zhang, X. Yang, M. Zhang, Y. Sun, and K. Long, "A genetic algorithm-based task scheduling for cloud resource crowd-funding model," *Int. J. Commun. Syst.*, vol. 31, no. 1, p. e3394, 2018.

[18] F. Tao, C. Li, T. W. Liao, and Y. Laili, "BGM-BLA: A new algorithm for dynamic migration of virtual machines in cloud computing," *IEEE Trans. Services Comput.*, vol. 9, no. 6, pp. 910–925, Nov./Dec. 2016.

[19] F. Tian, R. Zhang, J. Lewandowski, K. M. Chao, L. Li, and B. Dong, "Deadlock-free migration for virtual machine consolidation using chicken swarm optimization algorithm," *J. Intell. Fuzzy Syst.*, vol. 32, no. 2, pp. 1389–1400, 2017.

[20] S. Srikantaiah, A. Kansal, and F. Zhao, "Energy aware consolidation for cloud computing," *Scenario*, p. 10, 2008. [Online]. Available: http://www.usenix.org/events/hotpower08/tech/full_papers/srikantaiah/srikantaiah_html/main.html

[21] S. Goswami and A. Das, "Optimization of workload scheduling in computational grid," in *Proc. 5th Int. Conf. Frontiers Intell. Comput., Theory Appl., Adv. Intell. Syst. Comput.*, vol. 516, 2017, pp. 417–424. [Online]. Available: http://link.springer.com/10.1007/978-981-10-3156-4
[22] Q. Zheng, J. Li, B. Dong, R. Li, N. Shah, and F. Tian, "Multi-objective optimization algorithm based on BBO for virtual machine consolidation problem," in *Proc. Int. Conf. Parallel Distrib. Syst. (ICPADS)*, Jan. 2016, pp. 414–421.
[23] M. Cardosa, M. R. Korupolu, and A. Singh, "Shares and utilities based power consolidation in virtualized server environments," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, Jun. 2009, pp. 327–334.
[24] R. W. Ahmad, A. Gani, S. H. A. Hamid, M. Shiraz, A. Yousafzai, and F. Xia, "A survey on virtual machine migration and server consolidation frameworks for cloud data centers," *J. Netw. Comput. Appl.*, vol. 52, pp. 11–25, Jun. 2015.
[25] K. Li, "Scheduling parallel tasks with energy and time constraints on multiple manycore processors in a cloud computing environment," *Future Gener. Comput. Syst.*, vol. 82, pp. 591–605, May 2018. doi: 10.1016/j.future.2017.01.010.
[26] T. Kaur and I. Chana, "GreenSched: An intelligent energy aware scheduling for deadline-and-budget constrained cloud tasks," *Simul. Model. Pract. Theory*, vol. 82, pp. 55–83, Mar. 2018. doi: 10.1016/j.simpat.2017.11.008.
[27] K. Li, "Energy constrained scheduling of stochastic tasks," *J. Supercomput.*, vol. 74, no. 1, pp. 485–508, 2018.
[28] S. Midya, A. Roy, K. Majumder, and S. Phadikar, "Multi-objective optimization technique for resource allocation and task scheduling in vehicular cloud architecture: A hybrid adaptive nature inspired approach," *J. Netw. Comput. Appl.*, vol. 103, pp. 58–84, Feb. 2018. doi: 10.1016/j.jnca.2017.11.016.
[29] H. Liu, H. Jin, X. Liao, C. Yu, and C.-Z. Xu, "Live virtual machine migration via asynchronous replication and state synchronization," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 12, pp. 1986–1999, Dec. 2011.
[30] U. Deshpande, B. Schlinker, E. Adler, and K. Gopalan, "Gang migration of virtual machines using cluster-wide deduplication," in *Proc. 13th IEEE/ACM Int. Symp. Cluster, Cloud, Grid Comput. (CCGrid)*, May 2013, pp. 394–401.
[31] J. Zhang, E. Dong, J. Li, and H. Guan, "MigVisor: Accurate prediction of VM live migration behavior using a working-set pattern model," in *Proc. 13th ACM SIGPLAN/SIGOPS Int. Conf. Virtual Execution Environ.*, 2017, pp. 30–43.
[32] S. Zou *et al.*, "VirtualKnotter: Online virtual machine shuffling for congestion resolving in virtualized datacenter," *Comput. Netw.*, vol. 67, pp. 141–153, Jul. 2014. doi: 10.1016/j.comnet.2014.03.025.
[33] E. Casalicchio, D. A. Menascé, and A. Aldhalaan, "Autonomic resource provisioning in cloud systems with availability goals," in *Proc. ACM Cloud Auto. Comput. Conf. (CAC)*, 2013, p. 1. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2494621.2494623
[34] C. Liu, K. Li, Z. Tang, and K. Li, "Bargaining game-based scheduling for performance guarantees in cloud computing," *ACM Trans. Model. Perform. Eval. Comput. Syst.*, vol. 3, no. 1, pp. 1–25, 2018.
[35] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Generat. Comput. Syst.*, vol. 28, no. 5, pp. 755–768, 2012. doi: 10.1016/j.future.2011.04.017.
[36] M. Duggan, J. Duggan, E. Howley, and E. Barrett, "A network aware approach for the scheduling of virtual machine migration during peak loads," *Cluster Comput.*, vol. 20, no. 3, pp. 2083–2094, 2017.
[37] X. Chen, J.-R. Tang, and Y. Zhang, "Towards a virtual machine migration algorithm based on multi-objective optimization," *Int. J. Mobile Comput. Multimedia Commun.*, vol. 8, no. 3, pp. 79–89, 2017. [Online]. Available: http://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/IJMCMC.2017070106
[38] S. B. Melhem, A. Agarwal, N. Goel, and M. Zaman, "Markov prediction model for host load detection and VM placement in live migration," *IEEE Access*, vol. 6, pp. 7190–7205, 2017.
[39] A. S. Sofia and P. GaneshKumar, "Multi-objective task scheduling to minimize energy consumption and makespan of cloud computing using NSGA-II," *J. Netw. Syst. Manage.*, vol. 26, no. 2, pp. 463–485, 2017.
[40] R. Yadav, W. Zhang, H. Chen, and T. Guo, "MuMs: Energy-aware VM selection scheme for cloud data center," in *Proc. Int. Workshop Database Expert Syst. Appl.*, 2017, pp. 132–136.
[41] R. Yadav and W. Zhang, "*MeReg*: Managing energy-SLA tradeoff for green mobile cloud computing," *Wireless Commun. Mobile Comput.*, vol. 2017, Dec. 2017, Art. no. 6741972.
[42] R. Yadav, W. Zhang, O. Kaiwartya, P. R. Singh, I. A. Elgendy, and Y.-C. Tian, "Adaptive energy-aware algorithms for minimizing energy consumption and SLA violation in cloud computing," *IEEE Access*, vol. 6, pp. 55923–55936, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8479359/
[43] C. Clark *et al.*, "Live migration of virtual machines," in *Proc. 2nd Conf. Symp. Netw. Syst. Design Implement. (NSDI)*, 2005, pp. 273–286. [Online]. Available: http://dl.acm.org/citation.cfm?id=1251203.1251223
[44] T. Hirofuchi, H. Ogawa, H. Nakada, S. Itoh, and S. Sekiguchi, "A live storage migration mechanism over WAN for relocatable virtual machine services on clouds," in *Proc. 9th IEEE/ACM Int. Symp. Cluster Comput. Grid (CCGRID)*, May 2009, pp. 460–465.
[45] K. Ye, X. Jiang, D. Huang, J. Chen, and B. Wang, "Live migration of multiple virtual machines with resource reservation in cloud computing environments," in *Proc. IEEE 4th Int. Conf. Cloud Comput. (CLOUD)*, Jul. 2011, pp. 267–274.
[46] H. Liu, H. Jin, C.-Z. Xu, and X. Liao, "Performance and energy modeling for live migration of virtual machines," *Cluster Comput.*, vol. 16, no. 2, pp. 249–264, 2011.
[47] A. Verma, P. Ahuja, and A. Neogi, "pMapper: Power and migration cost aware application placement in virtualized systems," in *Middleware* (Lecture Notes in Computer Science: Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 5346. Berlin, Germany: Springer, 2008, pp. 243–264.
[48] W. Zhang, S. Han, H. He, and H. Chen, "Network-aware virtual machine migration in an overcommitted cloud," *Future Gener. Comput. Syst.*, vol. 76, pp. 428–442, Nov. 2017. doi: 10.1016/j.future.2016.03.009.
[49] A. Beloglazov, R. Buyya, Y. C. Lee, and A. Zomaya, "A taxonomy and survey of energy-efficient data centers and cloud computing systems," *Adv. Comput.*, vol. 82, pp. 47–111, 2011.
[50] F. Zhang, X. Fu, and R. Yahyapour, "CBase: A new paradigm for fast virtual machine migration across data centers," in *Proc. 17th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput. (CCGRID)*, May 2017, pp. 284–293.
[51] X. Meng, V. Pappas, and L. Zhang, "Improving the scalability of data center networks with traffic-aware virtual machine placement," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.
[52] V. Mann, A. Kumar, P. Dutta, and S. Kalyanaraman, "VMFlow: Leveraging VM mobility to reduce network power costs in data centers," in *Networking* (Lecture Notes in Computer Science: Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 6640. Berlin, Germany: Springer, 2011, pp. 198–211.
[53] H. Al-Aqrabi *et al.*, "VMScatter: Migrate virtual machines to many hosts," in *Proc. ACM SIGPLAN Notices*, 2013, vol. 48, no. 7, pp. 63–72.
[54] T. Yang, Y. C. Lee, and A. Y. Zomaya, "Energy-efficient data center networks planning with virtual machine placement and traffic configuration," in *Proc. Int. Conf. Cloud Comput. Technol. Sci. (CloudCom)*, Dec. 2014, pp. 284–291.
[55] L. Adhianto *et al.*, "OpenStack Neat: A framework for dynamic and energy-efficient consolidation of virtual machines in OpenStack clouds Anton," *Concurrency Comput. Pract. Exper.*, vol. 22, no. 6, pp. 685–701, 2010.
[56] R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and simulation of scalable cloud computing environments and the CloudSim toolkit: Challenges and opportunities," in *Proc. Int. Conf. High Perform. Comput. Simulation (HPCS)*, 2009, pp. 1–11.

XIJIA ZHOU received the B.S. degree in computer science and technology from Jiangxi Normal University, Nanchang, China, in 2016. She is currently pursuing the master's degree with the Department of Information Science and Engineering, Hunan University, Changsha, China. Her research interests include cloud computing and virtual machine migration.

KENLI LI (SM'16) received the Ph.D. degree in computer science from the Huazhong University of Science and Technology, China, in 2003. He was a Visiting Scholar with the University of Illinois at Urbana–Champaign, from 2004 to 2005. He is currently the Dean and a Full Professor of computer science and technology with Hunan University and the Director of the National Supercomputing Center, Changsha. He has published more than 160 research papers in international conferences and journals such as the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the IEEE TRANSACTIONS ON SIGNAL PROCESSING, the *Journal of Parallel and Distributed Systems*, ICPP, and CCGrid. His current research interests include parallel computing, high-performance computing, and grid and cloud computing. He is an Outstanding Member of CCF. He serves on the Editorial Board of the IEEE TRANSACTIONS ON COMPUTERS.

CHUBO LIU received the B.S. and Ph.D. degrees in computer science and technology from Hunan University, China, in 2011 and 2016, respectively. He has published more than eight papers in journals such as the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the IEEE TRANSACTIONS ON CLOUD COMPUTING, the *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, the *Future Generation Computer Systems*, and the *Theoretical Computer Science*. His research interests include modeling and scheduling of distributed computing systems, approximation and randomized algorithms, game theory, and grid and cloud computing.

KEQIN LI (F'15) is currently a SUNY Distinguished Professor of computer science with the State University of New York at New Paltz. He is also a Distinguished Professor of Chinese National Recruitment Program of Global Experts (1000 Plan) at Hunan University, China. He was an Intellectual Ventures endowed Visiting Chair Professor with the National Laboratory for Information Science and Technology, Tsinghua University, Beijing, China, from 2011 to 2014. He has published more than 550 journal articles, book chapters, and refereed conference papers. His current research interests include parallel computing and high-performance computing, distributed computing, energy efficient computing and communication, heterogeneous computing systems, cloud computing, big data computing, CPU-GPU hybrid and cooperative computing, multicore computing, storage and file systems, wireless communication networks, sensor networks, peer-to-peer file sharing systems, mobile computing, service computing, the Internet of things, and cyber-physical systems. He has received several best paper awards. He is currently serving or has served on the Editorial Boards of the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON CLOUD COMPUTING, the IEEE TRANSACTIONS ON SERVICES COMPUTING, and the IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING.